Class Notes

Lancer Brown

2/28/2020

Descriptive analysis

Goal Describe or summarize a set of data

1. Early analysis when receive new data
2. Generate simple summaries about the samples and their measurements
   - Eg: measures of central tendency or measures of variability
3. NOT for generalizing the results of the analysis to a larger population or trying to make conclusions

Exploratory analysis

Goal Examine the data and find relationships that weren't previously known

1. Explore how different variables might be related
2. Useful for discovering new connections
3. Help to formulate hypotheses and drive the design of future studies and data collection

Correlation does not imply causation!

Inferential analysis

Goal Use a relatively small sample of data to say something about the population

1. Provide your estimate of the variable for the population and provide your uncertainty about your estimate
2. Ability to accurately infer information about the larger population depends heavily on sampling scheme

Predictive analysis

Goal Use current and historical data to make predictions about future data

1. Accuracy in predictions is dependent on measure the right variables
2. Many ways to build up prediction models with some being better or worse for specific cases,
   - More data and a simple model generally performs well at predicting future outcomes

Just because one variable may predict another, it does not mean that one causes the other

Causal analysis

Goal See what happens to one variable when we manipulate another variable

1. Gold standard in data analysis
2. Often applied to the results of randomized studies that were designed to identify causation
3. Usually analysed in aggregate and observed relationships are usually average effects

Mechanistic analysis

Goal Understand the exact changes in variables that lead to exact changes in other variables

1. Applied to simple situations or those that are nicely modeled by deterministic equations
2. Commonly applied to physical or engineering sciences
   - Eg: Biological sciences, are far too noisy to use mechanistic analysis
3. Often the only noise in the data is measurement error

Experimental design

Formulate question -> Design -> ID problems and source error -> collect data

Hypothesis: what is outcome of experiment

Y-axis: Dependent variables X-axis: Independent variables

Example: diet effect on BMI etc. (Diet = dependant) Confounder: extraneous that could effect relationship dependent vs independent

Reduce confounding effects: controls, blinded, randomization

Replications: repeating experiment with multiple subjects.

p-value: probability results were observed by chance (p<0.05 - significant result)

p-hacking: exhaustively search for correlations in data in large data sets.

P console Input and Evaluation x <- 1 assignment operator (x is 1) print(x)

# indicates comment

x <- ## nothing printed

x ## auto-printing occurs

print(x) ## explicit printing

```
x <- 1
print(x)
```

```
## [1] 1
```

```
msg <- "hello"
msg
```

```
## [1] "hello"
```

```
## : is used to create integer sequences
x <- 1:20
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

Data Types

5 Basic (atomic) classes of objects - character - numeric - integers - complex - logical (True/False)

Most basic object is vector vector can only contain objects of same class - Cannot Mix ONe exception is list: which is a vector but can contain different classes

Empty vectors can be created with vector()

Numbers

- Numbers are numeric objetcs (double precision real numbers)
- If you explicitly want an integer, you need to specify L suffix.
- Ex. Entering 1 gives you a numeric, entering 1L gives you integer
- There is also a special number Inf which represents infinity: e.g. 1 / 0: Inf can be used in ordinary calculations: e.g. 1 / Inf is 0
- The NaN value represents undefined value "Not a Number" e.g. 0/0; NaN can also be thought of as a missing value (more on that later)

Attributes

R objects can have attributes

- names, dimnames
- dimensions (e.g. matrices, arrays)
- class
- length
- other user-defined attributes/metadata

Attributes of an object can be accessed using the attributes() function.

R Data Types - Vectors and Lists

The c() function can be used to create vectors of objects

```
x <- c(0.5, 0.6) ## numeric
x <- c(TRUE, FALSE) ## logical
x <- c(T, F) ## logical
x <- c("a", "b", "c") ## character
x <- 9:29 ## integer
x <- c(1+0i, 2+4i) ## complex

## Using the vector() function
x <- vector("numeric", length = 10)
x
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0
```

Mixing Objects

What about the following?

```
y <- c(1.7, "a") ## character - first element is 1.7, second element is "a"

y
```

```
## [1] "1.7" "a"
```

```
y <- c(TRUE, 2) ## numeric - TRUE gets converted to a number (TRUE = 1; FALSE = 2); vector 1,2

y
```

```
## [1] 1 2
```

```
y <- c("a", TRUE) ## character
```

When different objects are mixed in a vector, coercion occurs so that every element in the vector is of the same class. Coercion goes with least common denominator.

Explicit Coercion

Objects can be explicitly coerced from one class to another using the as.* functions, if available.

```
x <- 0:6
class(x)
```

```
## [1] "integer"
```

```
as.numeric(x)
```

```
## [1] 0 1 2 3 4 5 6
```

```r
as.logical(x)
```

```
## [1] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```r
as.character(x)
```

```
## [1] "0" "1" "2" "3" "4" "5" "6"
```

Nonsensical coercion results in NAs

```r
x <- c("a", "b", "c")
as.numeric(x)
```

```
## Warning: NAs introduced by coercion
```

```
## [1] NA NA NA
```

```r
as.logical(x)
```

```
## [1] NA NA NA
```

```r
as.complex(x)
```

```
## Warning: NAs introduced by coercion
```

```
## [1] NA NA NA
```

Lists

Lists are a special type of vector that can contain elements of different classes. Lists are very important data type in R and you should get to know them well

```r
x <- list(1, "a", TRUE, 1+4i)
```